The DOTWRITER 4.0

LETTERSET DESIGN SYSTEM

This book explains how to use the LETTERSET DESIGN SYSTEM (LDS) of DOTWRITER (tm) on the TRS-80 (R) micro-computer. It applies to LDS Version 4.0 and above. Additional Supplement sheets may be issued from time to time.

Date: March, 1984 Printing: 6

Published by:

PROSOFT, a division of TTSC
Box 560
North Hollywood, CA 91603
(818) 764-3131

ACKNOWLEDGEMENTS

DOTWRITER and the Letterset Design System were designed and written by William K. Mason. This book was written by Richard C. McGarvey, and printed entirely with DOTWRITER 4.0 on a C. ITOH 8510A. If your printer has the necessary mechanical capabilities and is supported by DOTWRITER, and you have the character fonts we used here, you can do anything you see in this book.

Cornucopia Software's "Electric Webster" spelling checker was used to check and correct most of the spelling throughout this book, and to generate hyphenation points. When using big fonts, we recommend you use hyphenation also, for improved readability.

TABLE OF CONTENTS

THE LETTERSET DESIGN SYSTEM FOR DOTWRITER 4.0

The Letterset Design System (LDS) is intended for use with the DOTWRITER Graphics Text Formatter. DOTWRITER prints previously written text as explained in the book that accompanies that program. The shapes of the characters it prints are defined in files called "Lettersets." Each letterset contains the dot graphics patterns for up to 95 symbols: numbers, upper and lower-case letters, and special symbols such as periods, dollar signs, brackets, etc.

The Letterset Design System is used to design those patterns, enter them into the computer, and then modify them until they meet your functional requirements and are artistically pleasing to the eye. LDS consists of two major components: "Tiny Graphics Editor And Programmer" (TGEAP), and "Letterset Manipulation Utilities" (LSMU). This book will show you how to use each of the features of these programs.

If you just want to use lettersets other people have designed, you don't need LDS: PROSOFT offers over 120 lettersets (as of early 1984), and others are in development. However, most people who use DOTWRITER are creative, and find that they want to "improve" our stock lettersets, tailor them to their own requirements, or design new lettersets of their own. Besides the practical benefits of having custom lettersets, many of us find great satisfaction in creating artistic designs. It's even more enjoyable to find that the design effort only has to be done once, since the computer (with DOTWRITER's help) will faithfully print your designs forever afterwards -- with no duplication of effort on your part.

If any of the reasons in the preceding paragraph struck a responsive chord for you, then you've made a wonderful decision in buying this package: minor changes to existing lettersets are trivially easy with the TGEAP drawing program; creating new lettersets is greatly simplified with TGEAP; and refining, combining, and changing the sizes and positions of symbols is simple, automatic, and fast with the LSMUs.

DOTWRITER supports two categories of printers: those that print eight vertical dots at a time (Epson and C.Itoh), and those that print seven vertical dots at a time (Radio Shack and Okidata). The differences among these four brands of printers are large enough to have forced us to have several different versions of DOTWRITER, and also several different versions of the Design programs. So, the LDS disk you've received should either be for 8-dot printers or 7-dot printers, and should match your printer. If you have the other kind of printer, return the disk to us immediately for exchange, since nothing you draw with the wrong disk can possibly work with your printer!

Before you can use LDS, you must copy all the programs from the distribution diskette to disks of your own. You can make as many backup copies of those programs as you need to feel safe, but the distribution diskette itself is in a format that may not work with your DOS "BACKUP" command. The diskette is in single-density, 35-track format, and is a "flippy" disk. That means there are programs on both sides, and to get to the second side, you just have to turn the disk over. Think of it as being two separate disks.

If you have a Model III, you'll have to "CONVERT" both sides to use them. Once you've done that, please COPY each program to your own disks. No further installation is required after that, so you can get started almost immediately.

The rest of this book is divided into two main sections: one for TGEAP and the other for the Letterset Manipulation Utilities. Each of those sections includes special installation considerations for its own programs. TGEAP is usually on the second side of the distribution disk, along with a few of the LSMU's. Most of the LSMU's are on the first side of the disk. We suggest you learn TGEAP first, and practice with it on a <u>COPY</u> of one of the stock lettersets.

<u>A WORD ABOUT DESIGNING LETTERSETS</u>

Like all skills, letterset design is something that must be learned. The programs are all easy to learn and use, so you'll become proficient with them quickly. You'll also find it very easy to "touch up" our lettersets. However, when you first try to design letters from scratch, you probably will not be totally pleased with your initial efforts. As you continue to design letters, your skill, and your understanding of the factors involved, will improve. Remember, <u>someone</u> designed the beautiful lettersets you've seen in our print samples, and that person didn't do "CAMEO" as his first effort.

Each of the five kinds of printers we support has a different number of dots per horizontal inch. The more dots there are per inch, the narrower a character will be, unless you use more dots. Also, the more dots there are per inch, the finer the shapes can be. Of the supported printers, the C. ITOH has the best dot resolution: 160 per inch. The Okidata has about 105, and the DMP's have about 60. The Epson, with 120, falls in the middle. You'll want to take these differences into consideration when designing your own character sets.

MODEL 4 CONSIDERATIONS

We don't offer LDS for use in native Model 4 mode because the screen graphics in that mode are uneven, which makes it impossible to design shapes that will match what gets printed. You can see this clearly by going into BASIC, and then typing this statement:

```
LPRINT CHR$(129);CHR$(132);CHR$(144)
```

That will produce three rectangles on the screen, in a staircase pattern. If you do it on a Model I or III, or in Model III mode on a Model 4, all three rectangles will be the same size and shape. However, if you do it in Model 4 mode, the third rectangle will only be half the height of the other two.

If you have a Model 4, and want to use LDS, do not despair: you can run them in Model III mode on your computer, where the graphics are all the same size, and then use the results in Model 4 mode!

With these preliminaries out of the way, let's find out how to use TGEAP...

INTRODUCTION TO TINY GEAP (TGEAP)

"GEAP" stands for "Graphics Editor and Programmer." It was the original on-screen drawing and printing system from which today's advanced DOTWRITER and TGEAP evolved. TGEAP allows you to modify any of our lettersets, or create lettersets of your own. It is easy to use and its prompts will guide you along. This chapter will show you how to install TGEAP (make it operational on your computer and operating system) and how to use it effectively.

Version 4 of TGEAP is written entirely in the "machine language" of the TRS-80, so it runs as a DOS command ("TGEAP"), and is both fast and smooth to operate. Since it is intended for use in designing letters, rather than for printing documents, it runs independently of the rest of DOTWRITER.

INSTALLING TGEAP

Before using TGEAP or LDS, be sure to transfer all the programs from the original disk to disks of your own. Thereafter, only work with your own copies, never with the original. The format of our distribution diskette was covered in the first section of this book.

TGEAP supports more than one brand of printer, so after copying the programs to your own disk, you should select the copy that applies to the printer you are using. Throughout this chapter, we'll call the program "TGEAP", and recommend you rename the appropriate program to "TGEAP/CMD", after which you can "Kill" (delete) the unneeded versions. The supplied versions are as follows, and only one or two of them will be on your disk:

```
TGEPS/CMD  - EPSON MX, RX, FX SERIES
TGITO/CMD  - C. ITOH 8510,1550
TGOKI/CMD  - MICROLINE 84,92,93
TGDMP/CMD  - RADIO SHACK DMP 200,400,500,2100
```

TINY GEAP Overview

TINY GEAP is designed to make it practical to design lettersets and drawings. It has 15 commands, most of which are used only once in any operation. There is also an instruction menu that lists all of the commands. This menu can be displayed at any time without interfering with drawings on the screen. The commands are all done with just one key each (for example, "S" stands for "SAVE"), so you can see why the program will be simple to use.

Let's take a look at the operation of TINY GEAP: TINY GEAP is used to create graphics that can be printed in HIGH RESOLUTION (dot-image graphics). Each symbol can be a letter or a drawing taking up one or more screens.

The operation is simple. Once a working frame is defined (a simple matter of specifying the dot-width and line height needed for your graphic) you simply move the cursor around in the frame, leaving lines and dots (picture elements, often called "pixels") where you desire.

Once you have drawn your graphic, TINY GEAP will translate each screen "pixel" into a printer dot.

The resulting graphic can be written ("W") to the printer for a test, edited until it is just right, and then saved to a disk file for future use. In this fashion, an entire letterset can be created!

When the letterset is complete, it can be added to your collection and used with Dot Writer at any time!

TGEAP has a Transfer command ("T") that lets you load any pre-existing letter, from any letterset, right onto the screen, and examine or modify it. The result can be placed back into the original letterset or into a new one. This feature allows you to start with a STOCK letterset and create a new one of your own.

Running TINY GEAP

Working with a BACKUP disk, Boot your DOS. Depending on the Disk Operating System you are using, you will have some type of DOS READY prompt. Then, just type:

TGEAP (and press the ENTER key)

When you do this, the TGEAP LOGO will appear on the screen. The logo will be displayed for a few seconds while the rest of the program is loaded into memory by DOS.

Finally, the logo will disappear and the screen will display the command summary that is shown in figure 2 (Appendix A). This command summary contains all you need to run the TINY GEAP programs. Once you've started to draw, the "L" command can be used at any time to re-display the command summary without disrupting your screen work. Now let's examine the commands!

Creating Characters with TGEAP

Note that the last line of the command summary says "Hit <ENTER> to continue". Hit the ENTER key now. The screen will clear and you will have a flashing " - " (minus sign) in the upper left corner of the screen. This signifies that you're in command mode. Whenever you are in this mode, you can access any of the commands on the command summary chart, or move the cursor (the minus sign) around without changing any part of your drawing. Bear that in mind, because it will come in handy when you have a complex figure on the screen and are looking for an easy way to move to another part of it.

To re-assure yourself that the command menu is still available, press the "L" key and watch as the command summary returns. Now hit the ENTER again and return to the command mode and blank screen.

I'm going to briefly review each command in the order they appear. When the review is finished, we will run through a creation session. You'll be surprised at how simple the operation is.

The Save Command: "S"

First is the "S" - save figure to disk. When you've finished editing, creating, or manipulating a figure, you can save it back to disk with this command. You must be in the command mode (flashing minus) to use this, or any command. If you're in "draw" mode, which uses a small rectangular cursor, simply press the minus key to switch back.

There are several things you will need to know about this command. First, it saves whatever is within your screen frame to disk. Second, it saves it under a keyboard symbol. Third, each keyboard symbol has a special storage area on the disk. Finally, the images are saved into a disk file.

The keyboard is arranged in ASCII order and so is the letterset disk file. the "!" is the lowest ASCII value and the lower-case "z" is the highest. The book that came with your computer has a chart of all the ASCII characters in the back, and you can refer to it if necessary. (There are differences between the Models I and III.) Since you can create and change letters and symbols in any order you like, the sequence is not too important. However, the "range" (lowest and highest) is very important, since it determines how much disk space the letterset will take. If your letterset doesn't have any lower-case characters, you can save considerable space.

The "S" command saves in ASCII order so that if you save a graphic under the "z" character, the entire file will be set up and space reserved for all ASCII values below "z". Since "z" is the highest keyboard ASCII value, room for all other characters will be reserved.

The reason for telling you all this is that, if you are creating an entire letterset, it doesn't matter in what order you save them, since you will always use all the spaces. On the other hand, if you are creating only one or two graphics, it is best to save them under the lowest ASCII value keyboard characters so as to conserve disk space.

Now let's look at the "S"ave procedure. Like all of the commands, "S"ave is self prompting. Once you have completed a graphic, press the minus key to return to the command mode. The flashing minus will appear. Now press the "S" key. The screen will clear and you will be asked for the keyboard character under which you want to store the graphic. If it is a picture and only a few will be in the file, answer with a low ASCII value character. If you are creating a letterset, answer with the corresponding letter. An "A" for A or a "c" for c, etc. Numbers, letters, special characters, it doesn't matter what you store under any given keyboard character. The only convention to follow is to match the keyboard with the proper character when creating a letterset.

Once you have responded to the last prompt, you will be asked to enter a filename for the character. If you have been working on a file, that name will appear as the default, otherwise, you can enter any filename that is valid under you DOS. When you respond to the filename prompt, the character will be saved and you will be returned to the command mode for your next function.

The Transfer Command: "T"

The next command on the menu is the "T"ransfer command. Again, you must be in the command mode (flashing minus) to enter any of these commands. If you are not in command mode, press the minus key. To make it easy to try this, and some other features as you read, we suggest you have a <u>copy</u> of a "stock" letterset on one of your disks right now.

The "T"ransfer command is designed to allow you to load a letter or graphic from a disk file for editing. Transfer automatically creates a frame for the loaded character so if you are working on creating a letterset, you do not have to remember the frame size. If you left off with the letter "J", just "T"ransfer that letter (or any other letter in the letterset) and the frame will be restored so that you can continue to create.

Here is the procedure. From the command mode (flashing minus), press the "T". You will be prompted for the keyboard character that you wish to enter. Answer with the letter that you want to load. You will then be asked for the filename of the file you want the letter to come from. Enter the filename. The letterset will then be opened, the letter will be loaded to the screen and the frame will be created. Now you can enter the graphic mode and edit or recreate the letter.

The Write Command: "W"

The next command is the "W"rite command. This command allows you to print, in high resolution dots, the character that is presently in the screen frame. Think of this as sort of a preview of the character. The character will be printed exactly the way DOTWRITER will print it later on. The only difference is that DOTWRITER can make it darker, and of course, use it within a word. TGEAP just writes the single character to the printer for inspection.

Once the character is in the frame and you are ready for a test printout, simply go to the command mode (press the minus key) and press the "W" key. Be sure that the printer is on. The character that is in the frame will dump to the printer in hi-res. If nothing happens, it may be because you didn't press the "W" firmly enough, or because the printer isn't "ready." When the "Write" command is recognized, the character will be erased from the screen temporarily as TGEAP transfers it. Afterwards, it will re-appear. The same thing will happen, by the way, when you use the "S"ave command to store any changes back on disk.

The Assign Frame Command: "A"

If you use a pre-existing letterset, the size of its characters will be defined for you. In fact, of course, that size was established by the person who designed the letterset in the first place. When using an existing letterset, you won't need to assign a frame size, but if you want to start from scratch, the feature is needed.

Let's try the "A"ssign command. While in the command mode (flashing minus), hitting the "A" will allow you to "A"ssign a frame size within which to work. You will be prompted for the number of dots wide you want the frame. This will correspond directly to the number of individual pixels wide the frame will be. It will also be the number of printer dots wide the letter or graphic will be. You should carefully think this decision out. Experiment with different widths to find the best one for the letter or graphic you're drawing. There are some pointers that will help you decide on a frame size later on in this section. A single letterset can only contain one frame size!

The next prompt is for the number of printer lines. Each line is eight (8) dots high. If you select 1 printer line, you will have eight dots high to work in. If you select 2 you will have 16 dots high, etc. This is a printer limitation and not a program limitation. The printer only uses 8 of the nine dots in the vertical parameter and you must program all eight. Look in the following section on pointers for some more information on this point. If you have a "7-dot" printer, (Microline or Radio Shack), the version of TGEAP you're using will adjust for the differences, but you may want to examine early results in case you want to change anything.

Frame Size Considerations

The screen size of the TRS-80 allows you to create a maximum frame size of 128 dots wide by 6 lines high. This means that a single character cannot be larger than 128 by 6. Now, 6 lines high is one inch, and DOTWRITER can magnify letters until each one is a full page high!

You can always make the graphic or letter smaller than the frame but never larger. If you are making a graphic drawing, you need only make the frame large enough to hold the graphic - it doesn't hurt to go too big.

If you are creating a letterset, you will want to use the smallest frame possible and unless you also have the Letterset Manipulation Utilities you cannot place the created letters into a smaller frame. In order to figure out how to select the width, we suggest that you try creating a large letter first. Try creating an "M" for width. The "M" should fill the frame. The other letters will then fit - width wise.

You might also want to consider the style of the letters. If you are creating a fairly common letterset, the "M" will be fine but if you plan to flare the letters, the "W" might be a better choice for the width selection.

Height and lower-case are other factors. You will have to allow for descenders if you plan to have them. This can be done in several ways. Once you have selected a width, you must decide how much room you need to complete a descended character. Use a "g" or "q" for this purpose. Again, the style of the letter may need to be taken into consideration.

Remember, you must select height in increments of eight (8), so think ahead. If you have selected 1 printer line (8 dots) and create your uppercase letters so that they touch the bottom of the frame, you will not be able to have lower-case descenders that fall below the baseline of the uppercase. This means you must create your uppercase enough dots above the bottom of the frame to allow for the descenders! Eight dot high characters print faster than multi-line lettersets because they require only one pass. Our MicroPrint and Clarity are examples of the 8 dot high lettersets.

Finally, I mentioned the Letterset Manipulation Utilities once before. These utilities allow you to switch letters into different frames (among other things). If you have these utilities, you can make a frame larger than you need, create the letterset, then use the utility to find out the minimum frame size you can use for the letterset. The utility will then create the new frame size and allow you to move your letterset from the larger to the smaller frame. You can also move sections of the letterset so that you can mix and match lowercase and uppercase letters. I advise these utilities for anyone who is serious about making lettersets. They are very reasonably priced and well worth the money. There are over a dozen utilities, and they are fully menu driven. If you just plan to use the many "stock" lettersets we offer, and not modify them extensively or do much development of your own, you really won't need "LSMU"; but if you find your artistic flair is keeping you at the computer beyond reasonable hours, they can save you a lot of time.

The Graphic Mode: "." (period)

Now, at long last, we will begin to describe how you can draw with TGEAP (after all, that's what it's for, isn't it?) To get into "drawing" mode, just press the "." (period) key. This command exits the command mode and enters the graphic mode. If you do not have a frame specified, you will be warned and asked to create a frame. If you have used the "F" command or the "T" the "." (period) will take you to the graphic creation page and you will see either the blank frame or the transfered character. While in this mode, you will see a flashing dot (single pixel) cursor within the frame. If you press the minus key, the cursor will change to the flashing minus; hitting the "." (period) will bring back the flashing dot.

The Frame Redraw Command: "F"

The next command is the "F"rame draw command. You will almost never use it. It simply redraws your original frame if you have erased it. The frame and its contents, can be erased by simply pressing the CLEAR key while you are in the graphic (flashing dot) mode. The entire screen will clear and you will be returned to the command mode (flashing minus). Pressing the "F" will redraw your frame so that you can continue to create letters.

__WARNING:__ Pressing "CLEAR" erases your drawing without saving it. Don't do that unless you want to start over, or have "S"aved the graphic to disk.

Determining Storage Size

The "H"ow big command is uses to determine storage size of a letterset. Once a frame has been created, simply go to the command mode and press the "H". The frame size will be evaluated and a display will show you how large your letterset will be, as well as the size of the frame now in use.

The maximum letterset size is 32767 bytes and some frame sizes can exceed this limit. That is the reason for the "H" command. If you were to specify a frame of 128 wide by 6 high, the "H" command would return the following display:

```
Current Letterset is  PL/PR
Letterset Width is 13
Number of Print Lines is 1
Storage Requirements For Current Letterset (bytes)
! to /    13056 Bytes
! to 9    19968 Bytes
! to Z    44544 Bytes
! to z    73728 Bytes
Maximum File length is 32767 Bytes
Hit <ENTER> to continue?
```

As you can see, the entire letterset will not fit in the disk file. It will also not fit into memory all at one time! It is up to you to decide how many and which characters you will use in this case. Most graphics will not give you any trouble and most lettersets will be small enough to fit into the disk file. If they don't, you must decide which ones you want. Use the "H"ow big command to decide where to quit. Just a note: you can't beat this problem by placing the first half of the alphabet in one file and the second in another. While the theory is good, the second half of the alphabet would still have to be stored under the first half keyboard characters since the file size is determined by the highest ASCII value character.

The Summary Display Command: "L"

You have already seen the "L"ist command is action. While in the command mode (flashing minus) and when the frame is displayed, typing the "L" will give you a display as in figure 2 (Appendix A). This is a command summary. The character or graphic will not be disturbed and you can press the "." (period) key to return to it.

The Exit Command: "E"

The "E" command is simple to use. While in the command mode (flashing minus), pressing the "E" will exit DOS. Since this terminates TGEAP, you will want to have "S"aved your last graphic to disk before using this command.

Using The Arrows

You can move either cursor (dash or dot) around by using the arrows. They can be used alone or in combination, to get diagonal movements. Diagonal movement is hard to control and not recommended for beginners.

Graphic Mode Commands

Now that you have seen how the command mode works, let's take a look at the graphic commands. This is very simple so I'll just run through them in a summary format:

Using the arrows will move the cursor (in this case the flashing dot) in the indicated direction. The cursor will stay within the frame and will "wrap around" when frame boundries are encountered.

Holding the <SHIFT> key while moving the cursor will turn ON a pixel; if you keep moving the cursor as you do this, a line will be left behind. This is how we draw the characters. If the cursor is moved when <SHIFT> is NOT pressed, any pixels it touches will be turned OFF. That lets you skip areas that should be blank, or correct errors.

If you want to move around a large frame, it's quicker and safer to go to the command mode (flashing minus) and move the cursor: that cursor doesn't erase anything.

While the flashing dot is present, hitting the <CLEAR> key will clear the ENTIRE screen. You can regain the frame by pressing the "F" key while in the command mode, but you will lose the graphic you were working on. As we said earlier, "be careful."

Pressing the minus sign, will return you to the command mode (flashing minus cursor). You will have to due this in order to use any of the command mode commands!

Finally, cursor speed is controlled by the numbers 0 - 7. The higher the number, the slower the cursor. Number 7 will cause the cursor to creep along and 0 is so fast that is is almost unusable. I generally use 4 or 5 for fine work.

That concludes the command summary and instruction. Most of you will be able to run the program now and will create letters without any difficulty. For those who are still a bit cloudy, the following section will walk you through a letter creation.

The Walk Through Example

Start TGEAP from DOS by typing its name, then press <ENTER>. When the command summary is displayed, hit the <ENTER> key. You will now have a blank screen with a flashing minus in the upper left corner of the screen.

For our example, we will need to create a frame, a letter and a file to play with. For now, press the "." (period) key. See the error message? That will happen whenever there is an attempt to do something that requires a frame and a frame has not yet been created. A frame can be created with the "A" command or an old frame can be loaded with the "T" command. Look back to refresh your memory on how these operate.

Let's create a frame: press "A". You will be asked for the number of dots wide. Use 15 for now. Next you will be asked for the number of printer lines high (each line has eight dots). Select 2 for now.

After you have answered the questions, the screen will clear, a frame will appear in the upper left corner and you will be back in the command mode. Now press the "." (period) key. The cursor changed to a dot. Use the ARROWS ONLY to move it around the frame. Also, try the 0-7 keys and watch how the speed is changed. Also note that the cursor won't leave the frame.

Now, hit the <CLEAR> key. The screen is cleared and the command mode cursor (flashing minus) is back. Anytime you want to clear the screen, this procedure will work. If you have specified a frame size, change your mind and specify a new frame size, both frames will be on the screen. Using the <CLEAR> will clear the screen and the "F" will restore the current (newest) frame. You can try creating multiple frames now but be sure that the last one you use is the 15 wide by 2 high. That way you will be in step with our example.

The screen is now clear and the flashing minus is on the screen. Press "F". The frame reappears. It's time to draw a letter. Make it simple. The letter "L" is nice for this. Remember that you will have to stay inside the frame and that you will be able to adjust the speed of the cursor. Press the "." (period) key and use the arrows and <SHIFT> to draw the lines.

You should now have something on the screen. It doesn't matter what. Press the minus key and watch the cursor change. You are now in the command mode. Be sure that your printer is on and press the "W" key. The character in the frame will print out. The result is how the character will always look. The screen will restore to normal and you can edit the character until it looks right. Play with it a while and see how it works.

I hope that by now you have created a character that you are satisfied with. Let's see how many of these characters you can store to disk. Check the command summary by pressing the minus key (enter command mode) and then press the "L" key. The commands will list. See the one we want? It is the "H" command. Press <ENTER> and go back to the screen. You will still be in the command mode. Press "H". The following information will display:

```
! to /  510  Bytes
! to 9  700  Bytes
! to Z  1740 Bytes
! to z  2880 Bytes
```

We have plenty of room. Actually, you only need to check this when you are working with large frames.

Now hit <ENTER> and return to your character. Next, let's save it to disk. Rather than calling it "L" or whatever letter you drew, we'll save it under the lowest ASCII value to save disk space. You should be in the command mode (flashing minus), if not, get there. Now press the "S" key for save. You will be prompted for the keyboard character you want to save your graphic under. Answer "!", without the quotes of course. This is the lowest ASCII value and will not reserve any disk space below it. The next prompt is for the filename. Use and valid file name. Try : TEST/LET:1. This will save the letter under the file name TEST with the extension LET. It will also put the file on the disk in drive 1. Feel free to alter this name in any way you want.

That's all there is to it! You have created and saved a letter. Let's load it back in now. For the sake of experimentation, reboot your computer to start from scratch. When the command menu is displayed, hit <ENTER> to get a blank screen in the command (flashing minus) mode. Press "F" - nothing happens. Press "." (period) and you get an error message, since no frame is defined.

Now press the "T" key. You will be asked for the keyboard character that you want to load. Type "!" (again, without the quotes). You will be prompted for the filename. Enter the filename that you used to save your character. The access to disk is made and the character reappears on the screen.

Just to be sure the frame is there, enter the graphic mode (press the period) and then clear the screen (press the <CLEAR>). Now press the "F" and you will see your frame reappear. No letter - we erased it.

Type "T" now and specify the "!" to the prompt. Note that the second prompt now has a default. It is the filename that you entered a few minutes ago. Press <ENTER> only and the letter will reload and redisplay.

That's all! It is that simple. If you want some fun, try loading in some of the characters from our supplied fonts. Look at them to see how they were made. Alter them and print them out (use "W") and see how they look. Remember not to permanently change an original font style, since you might regret it if you haven't made a backup copy.

For information on using the lettersets you create, see the DOTPRINT portion of the manual. For further information on manipulation of the fonts, see the Letterset Manipulation Utility manual.

Letterset Manipulation Utility Instructions

This section will show you how to use our advanced Graphic Manipulation routines. To use them effectively, you should be familiar, if not proficient, with TGEAP and Dotwriter itself. Although this documentation covers the graphics utilities fully, we assume you have a good understanding of how to use your computer and you DOS. We will document only our programs and not the various DOS's that you might care to run them under.

The 4.0 version Manipulation Utilities are machine languane programs. All of them are run by selecting them from a menu, and that menu keeps track of the files you are using. Since there are two kinds of printers (8-pin and 7-pin), there are two menus and two sets of utilities. The disk you've received has all the programs for ONE kind of printer on it. Before using any of the programs, COPY ALL OF THEM to your own disks, and work with those copies afterwards. The programs are listed below:

To use the utilities, just type the name of the menu program as a DOS command. That name is either "LMU8" or "LMU7". Once it starts running, the menu displays options for each of 13 utilities. Type the number of the utility followed by the ENTER key to load the utility you want. At the completion of each utility option you we be allowed to RERUN, GO TO MENU or END. These options are self explanatory.

NOTE THAT THE FOLLOWING LIST IS BASED ON THE 8 BIT PRINTERS. THE 7 BIT PRINTER VERSION WILL SIMPLY SUBSTITUTE THE NUMBER 7 FOR THE NUMBER EIGHT IN EACH NAME. THE PROGRAMS ARE NOT INTERCHANGEABLE! The following is a list of the names of the programs on the Letterset Manipulation disk. There are fourteen programs, one for each of the 13 functions and the LMU/CMD program which is the menu program:

```
1.  SPLIT8/CMD     6.  ITAL8/CMD      11.  ROT8/CMD
2.  COPY8/CMD      7.  JUST8/CMD      12.  MIN8/CMD
3.  FORMAT8/CMD    8.  MERGE8/CMD     13.  PROP8/CMD
4.  BLANK8/CMD     9.  MAGEX8/CMD     14.  LMU8/CMD
5.  DUMP8/CMD     10.  WIG8/CMD
```

RUNNING THE UTILITIES

The utilities should not be run by themselves, only from the menu. Always start with the command, "LMU8" or "LMU7". When you do so, this list will be displayed:

```
                    Letter Manipulation Utilities
1 split letter into pieces, magnify pieces
2 copy between lettersets
3 format new lettersets
4 list blanks in letterset to printer
5 list byte dump of letter(s) to printer
6 italicize letterset
7 move letter(s) within frame, and/or justify or center
8 merge / fill letters
9 expand or magnify letter(s) within frame
10 add wiggle to letter(s)
11 rotate and/or mirror image, shrink
12 find minimum storage values
13 make letterset proportional
<ENTER> number of your choice . . .
```

In order to select an option, simply type the number of the routine and hit <ENTER>. The desired routine will automatically load and run.

Now let's take a look at each routine and briefly examine its operation. Remember, we will expect that you are already familiar with the DOT WRITER / TGEAP graphic creation routines. We will not spend any time explaining those facilities.

CHOICE 1, THE SPLIT ROUTINE

The split routine will allow you to enlarge and refine a graphic in the following fashion. First, remember that each graphic is designed in a frame. For point of example we will use the full screen as the frame but, the actual frame can be any size from 1 dot wide by eight dots high, to 128 dots wide by 48 dots high. Also remember that, even though the width can be incremented by single dot units, the height must be incremented by printer lines where 1 line equals 8 dots high.

Once you have designed your graphic, be it a letter or a picture, you will assign it to a keyboard character. For our purposes we will always assign the original graphic to the "!" character. With this character saved under a filename (ALPHA for example) you can now run the SPLIT program.

When run, SPLIT will ask you for the file containing the graphic you wish to split (INPUT FILE). In this case, we reply ALPHA. You would answer with the file name containing the letter or graphic you want to split. Next, for the filename of the output file. It can be the same as input, but you'll destroy the original so be careful!

Now the SPLIT routine will display the parameters of the letterset file you are using. It is a good idea to record this information and file it. It will save you time cross checking in the future.

Next you'll be asked for the character to be split. Answer with the exclamation point "!" (don't include the quotes). And finally, you'll be asked if you want two generations. The first generation is a 4 time split and the second is a 16 time split.

Now the SPLIT program begins its work. You will see what is happening on the video display. As an aside, when we first started working with these programs they were all VERY slow. Some were so slow that you could start the routine and go out to dinner. When you came back, providing you're a slow eater, the routine would be finished. You will see however, that the routines are all very fast now and many occur so quickly they will be hard to follow.

The SPLIT program will first divide the graphic FRAME into four parts. Each of the four parts will then be magnified PROPORTIONALLY to fill the entire size of the original frame. The resulting four quarters, which are now each as big as the original, are stored under the keyboard characters ", #, $, %. No file creation is done at this point.

If you have asked for two generations, the procedure will be run again, otherwise, the new graphics will be stored under the above keyboard characters and placed in the file we named BETA (or whatever name you specified as the output file). Each part will be as large as the original and each can be loaded separately for editing. In order to print out the larger graphic, you would use DOTPRINT and would load file BETA with the .BF BETA command. Then, by printing the four sections in order (or 16 sections):

```
"#
$%
```

you will be able to recreate the original graphic only 4 or 16 times as large.

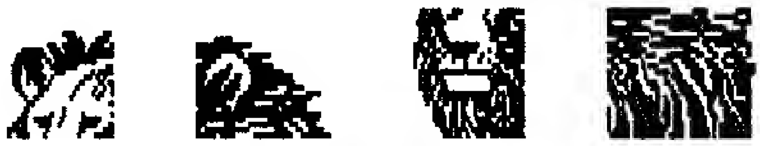
If you answer YES to the TWO GENERATIONS? question, the entire procedure will be rerun on each of the four quarters, making a 16 times larger diagram. The screen displays the layout of the characters that each segment will be saved under, in their relative positions, at the end of each split. The 16 time split is:

```
&'  *+
()  ,-
./  23
01  45
```

Now let's look at an example of how the SPLIT program works. First we start with our drawing in file ALPHA stored under character "!". I'll use the .BF command to load ALPHA.

This little lion was created on one screen of TGEAP. Now I'll SPLIT it into 4 parts and display each of the 4 parts separately and then together. This will be file BETA, and the characters will be " # $ and %. First I'll display them in that order, then in the proper order:

```
"#
$%
```



Note that each of the four parts is now the same size as the entire original graphic. You will also note that the first SPLIT of the graphic caused a stair step effect in the angled sides of the graphic. With a little editing with TGEAP, you can smooth out any rough edges.

Every magnification will increase the BLOCK appearance of the resulting graphic. Be sure to edit the result of the magnification to smooth out the lines and round out the edges.

The 16 time split works in exactly the same way. Refer to the diagram for the character layout. Also, when you use the 16 times split, the character edges will be even rougher than with the 4 times split. Again, each resulting character can be edited and this is an easy procedure since the split sections are proportional. All edges will continue to match up.

That is all there is to it. With the SPLIT routine you can easily create a graphic up to 16 screens in size. Now let's move on to the next routine.

In the discussion of Option #1, the SPLIT routine, I gave an example for you to follow. In most of the following sections I will not give you an example but will explain the use and prompting procedure.

While some routines like SPLIT and WIGGLE can be best explained by example, others, such as COPY LETTERS and FORMAT are too simple to demonstrate. For example, the FORMAT routine formats a file to your specifications. You tell it how many dots wide and how many printer lines high the file should be and it does the rest. There is really nothing to demonstrate.

CHOICE 2, THE COPY LETTER ROUTINE

This is one of the simple routines I spoke of. No glamour here! With COPY you can copy the letters (any one or group of letters) from one file to another. This makes it possible to take the lowercase characters from one font and place them into the position of the lowercase characters of another font. The only requirement is that the receiving file frame size must be large enough to hold the incoming letters. Don't get me wrong! You can copy a large letter into a small frame - but the result will be a chopped off letter.

Let's take a look at the procedure. First, get an ASCII character order chart. There is one in the manual that came with your computer. Remember that keyboard characters, regardless of what you have stored under them, are stored on the disk in ASCII order from the LOWEST value to the HIGHEST value. That means (look at the chart) that the "!" is the first character in the file and the "z" is the last!

Now, we'll select option #2 from the menu. You are prompted for the name of the file to be copied from: That is the INPUT FILE. Type in the name of your INPUT FILE (that's where you want the letters to come from). Next you will be asked the name of the file that the copied letters are to be sent to: That is the OUTPUT FILE. If you want to duplicate the uppercase characters into the lowercase positions you can have the same file for both INPUT and OUTPUT. Now for range of characters. In order to specify the range of characters, you must know what order characters are stored in. That means you must use your ASCII chart. For argument's sake we'll assume that we are going to transfer the lowercase of file B into the lowercase position of file A. The INPUT file is "B" - that is the source. The OUTPUT file is "A" - that is the destination.

Next, you will be asked for the LOWEST character to be copied. That means: What is the LOWEST (in ASCII value) character you want to take from the source (B) file? Since we are copying the lowercase, the answer would be??? Right! the answer is "a". Look at your chart and see that, of all the lowercase characters, "a" has the lowest ASCII value (97). You will further note that the highest ASCII value is "z" with a whopping 122. Isn't it convenient how they came out in order? So, for the LOWEST character to be copied you will respond "a".

Your next prompt will be for the HIGHEST character to be copied. Again, this refers to highest in ASCII value and of course, the answer in our example will be "z". Now we have specified the range. In fact, our range is the entire lowercase "a" to "z". If we had wanted to copy only one character we would simply specify the same character for both LOWEST and HIGHEST. Easy isn't it?

Now for the last step. All we have to do now is to tell the COPY routine where we want our range of characters copied to. We have given it the filename so we need only tell it where in that file to store the letters. We have also told the COPY program the range of letters (how many) we want to copy. All that is left is to tell the COPY program where to place the first (LOWEST) character. The final prompt is for the LOWEST character to be OUTPUT. The program is asking you for the location where it should start placing the range of letters we selected. Since we are copying the lowercase into the lowercase position, it would be a good idea to place the "a" into the "a" position. So the correct answer to this final prompt is "a". Now the program will do the rest. The "b" will be stored under "b", "c" under "c", and so on until the range is complete and the copy function is done.

Just suppose for a moment that we had answered "A" for the final prompt instead of "a". What would have happened? No, don't guess, because I'm going to tell you anyway. If we had made that error, the entire lowercase of file "B" would have been stored in the uppercase locations of file "A". When we typed out the file later on, the resulting text would be somewhat disrupted since all uppercase characters would appear as lowercase and all lowercase characters would be a different lowercase!

Not much to it is there? Well, don't be deceived! Once you begin using this package you will find this little COPY routine very valuable. Suppose you create a font of all fancy, uppercase letters. You have spent hours detailing and debugging this work of art. Now you sit down to type and you realize that you must ALWAYS remember that when you use this font, the entire text MUST be in uppercase. If you let a lowercase letter slip in you will not get a beautiful uppercase replacement. You will get garbage. Well, copy the uppercase into the lowercase positions dummy! There, one less thing to worry about.

Now let me briefly tell you of the foibles of the COPY program. First, if you copy a letter from a small frame size file into a large frame size file you will get a complete and accurate transfer. However, the small letter will be padded to fit the larger frame and will not necessarily look good with its new frame. The closer in frame size two lettersets are, the better they will look together. Second, if you transfer in the reverse direction, a large frame size into a smaller frame size, the larger letter may be cut off. I say "MAY BE CUT OFF" because a letterset does not absolutely have to be in an exact fit frame. If the larger letter is only a little larger, it may be OK. However, if you copy Olde English into the Microprint frame, I can guarantee you that the result will be garbage!

If the frame sizes of the intended transfers do not match, you will be prompted before the program continues. This prompt is informative only. If you have created a letterset in a large frame, checked the size (with option 12 of course) and found that you can use a smaller frame, you can create the smaller frame with option 3, and then copy the letters into the smaller file. You will get the warning prompt but all is OK.

What all of this boils down to is that you must know what you are planning to do BEFORE you do it. Experimentation is the name of the game and that is not a cop out. The entire reason for this package is so that you can experiment and manipulate. That is why we call it a MANIPULATION package.

CHOICE 3, THE FORMAT PROGRAM

FORMAT is a very simple routine to use. Let's say that you have created a letterset. You have checked it and found that the frame size you used is much larger than you need. All you have to do is FORMAT a new file, with smaller frame parameters, and then copy the letterset into the smaller file! Here is how it works.

Select option #3 from the menu. You will be asked for three things: The file name of the intended new file -

TEST/FIL.PASSWORD:2

You will be asked for the number of DOTS wide (you'll understand that better later on) 32, and the number of printer lines (each printer line is 8 dots high) 2. FORMAT will then create a new file of 32 dots wide by 16 dots (2 printer lines) high. Now you can copy the letterset into the new file. That's all there is to it. The thing to know about FORMAT is that it will allow you to create a file with dimensions you specify.

CHOICE 4, LISTING THE BLANKS IN A FILE

Another simple routine. You are prompted for the name of the object file. The program then checks that file and lists, TO THE PRINTER, the letter locations that are not yet filled. This routine comes in handy when you are creating a letterset and are forced to stop before completion. By running this program, you can find out which letters you have yet to do.

There is a problem with using this routine. It only works on a clean disk. If garbage has been saved into a letterset file, or if an aborted letter creation has been saved, that garbage or part letter will be counted as a letter. The best way to avoid this problem is, when you create a letter that you don't like and you want to try it again at a later date, save a BLANK frame to that key location on disk. The method for doing this is part of the TGEAP program.

CHOICE 5, THE BYTE DUMP ROUTINE

Again, this is a simple routine. The object is to dump a list of the actual byte layout of a graphic or range of letters. The operation is self prompting. You will be asked for the filename of the object file. Then you will be asked for the LOWEST letter and then the HIGHEST letter to be dumped. The range you specify is based on the ASCII table you have been supplied. If you wanted to dump a whole letterset, you would specify "!" as the LOWEST letter and "z" as the highest. If you only wanted to dump one letter, say the "B", you would specify that letter for both LOWEST and HIGHEST letters.

This particular utility is designed for the advanced user and has little practical application for the average user. One of our main uses is when we copyright a letterset. We copyright the BYTE DUMP as well as the appearance of the letter and the name. This gives us double protection.

The remaining utilities (with the exception of number 12) are relatively difficult to use. Rather than crash on through, I'll start a new section for the remainder of the utilities.

CHOICE 6, THE ITALICIZE ROUTINE

The italics option is relatively easy to use, however, it offers some unique challenges to the user. Let's first consider the prompt sequence.

When run, the italics option will first ask for the input file. This is the name of the file you wish to italicize. The second prompt is for the name of the output file. The output file can be the same as the input file BUT it is usually necessary to specify a new file due to different size requirements. I'll give you more on that later.

Next, the parameters of the input (source) file are listed. As I mentioned before, it is best to copy these and file them for future reference, especially if you are working with your own lettersets which have not been measured!

The next prompt asks you if you want to LEFT JUSTIFY the letters in the file. This may be important to do because the tilt added to the letters also adds width to the overall letter. If you don't left justify, and you keep the same frame size, the italicize routine WILL cut off the upper right side of the letters. The degree of the cut off depends on how wide the frame is and how much tilt is added.

You will also be asked for the lowest and highest values to be tilted. This allows you to italicize part of a letterset. If you have to create a larger frame size file (see FORMAT), you must also copy the untilted letters into it so that the fonts will fit in the same file.

Remember, files can only contain one frame size each and if you create a larger frame size for the italic letterset and intend to use that as an alternate letterset, you will have to specify the LARGER italicized font as the primary (BF) and the normal font and the secondary (AF) font. Then you will have to completely reverse the AF/BF commands. To boil that down: Since the italicized font is larger that its root font, it cannot always be specified as the alternate font. If you do, the DOTPRINT program will signal an error. So, you must specify the italic font with the BF command, the root font with the AF command and then, EDIT your text with all text EXCEPT that which is to be italics within the alternate font control codes.

Seems like the long way around doesn't it? That is why I suggest that you move the root font into the same size frame as the italic font. In that way the italic font can be specified with AF and used normally, as the alternate font AND, with the proportion routine, the additional width will not be a problem in the root font frame size!

OK, that out of the way, let's take a look at the italic routine itself. The final prompt will ask how much tilt you want for the italic font: 1 gives the most tilt and 32 the least. We have found that the tilt factor of 2 looks best with most letter quality fonts.

The best way to determine which tilt factor is best for your application is to try out a few. We can not supply steadfast rules in this case. Just experiment and take what you like! Now I want to take a minute to discuss frame size in more detail.

When you use the italic feature, the top dot pattern of each graphic will be shifted to the right by some amount that you will specify (tilt factor). The easiest way to avoid chopping off letters is to do the following:

1. Run option 5 or 12 to get the frame size (only the width is needed) of the root file. If you have listened to my earlier advise and filed the frame sizes when printed by other routines, you may have the size filed somewhere and you'll be able to skip this step. If not, be sure that you file the results of this run!
2. Now, add an obviously large frame width to that number. For example, if the frame width of the root file is 33, double it to 66. This will never pose a problem except in extremely wide fonts where you might hit the 128 max.
3. Use the FORMAT option and create a file that is wider than the root file but uses the same number of printer lines.
4. Next, use the copy routine to copy the root file into the larger file. You will also want to run the MOVE / JUSTIFY routine and place the letters top left in the frame!
5. When in the move / justify routine, left justify the new large file. I have also found that it saves trouble if you move all letters one dot right after the justification.
6. Now run the ITALIC routine until you get the look you want. When you are satisfied with the tilt of the letters, run option 12 and find out the smallest frame you can get your new font into. Then, create the new file and copy the italic font into it. And, as I mentioned earlier, copy the root font into a file of the same size!
7. Now run option 13 - the PROPORTIONAL routine on both of the fonts, the new root font and the italic font. This will create a new width table in the files.

That is all there is to it, and although there are shorter ways, this is the safest. When you use an italics letterset, it will look better if you turn on kerning ".KRON" and set the kerning space to at least two ".KS2".

Now onto the next routine. The following MOVE / JUSTIFY routine is a bit involved so be prepared to sit down at your computer and try to follow along. The routine isn't difficult but there is a lot to it!

CHOICE 7, MOVE / JUSTIFY

This utility lets you re-position a range of characters within their cells. You can move the characters up, down, left, right, or have them centered. It's menu-driven, so once you've answered the normal questions about Keyboard/Ascii entry, etc., you'll be shown the first of two menus:

```
Left-Right Shift -- Select by Number:
 1. Left Justify
 2. Right Justify
 3. Center
 4. Operator enters amount of shift
 5. No left/right shift
```

This menu-driven approach is new in Version 4, and the choices are self-evident. If you choose #4, you'll be asked which direction and how many dots; if you go too far, you will lose some dots from the character.

After you've answered the Left-Right menu, you'll be shown the Up-Down menu, which is a new feature of 4.0:

```
Up-Down Shift -- #
 1. Top Justify
 2. Bottom Justify
 3. Center
 4. Operator enters amount of shift
 5. No up/down shift
```

"Centering" on this menu refers to vertical centering, of course; on the left-right menu, it referred to horizontal centering.

After you've answered both menus, the entire range of characters in the input letterset will be re-positioned into the output letterset. The symbols will flash on the screen as this happens, and it moves along very fast.

Before we get into the next two utilities, I suggest you take a short break. "MERGE" is powerful, useful, and requires careful planning; "WIGGLE" is, well... different.

CHOICE 8, THE MERGE ROUTINE

OK! I've put this one off long enough. Here is the MERGE routine. To start with, a brief explanation of what the MERGE is for.

Basically, all the merge command does is ADD or SUBTRACT a specific pattern from each character in a letterset. If for example, you wanted a permanently underlined font, you could create the underline, save it to a file, then MERGE it into each character of the object letterset using the MERGE routine. Then, each character in the letterset (or and range thereof) would have the underline added to it. In reverse, you could use the MERGE command to remove the underline from the letterset.

FILL works in the same fashion, but it REPLACES the characters with the FILL pattern. REMEMBER, MERGE ADDS the pattern, FILL REPLACES the letter with the pattern.

How to do it? Well, for an example, I'll use the ROMAN/PR letterset. First, I'll use option 11 and shrink the letterset. This is just to give me a new letterset to work with. Then I'll center, vertically and horizontally, the new small letterset. It will look like this:

Roman Small Letterset

You won't be able to detect the centering in the above sample.

Now I'll run the TGEAP program and create a character to merge with every letter in the letterset. When in TGEAP, I just transfer a blank letter space (the up arrow position works well because it is almot never used) and I create a srting to merge. NOTE: If you are not sure of the amount of room you have, just TRANSFER in the largest letter in the letterset (usually M or W) and make your merge graphic around it (don't forget about descenders). Then Erase the parts that are not part of the merge graphic with the cursor.

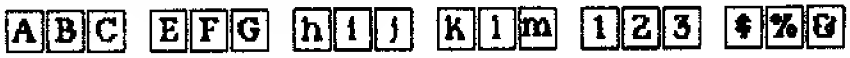
I created a box character to surround each letter. It looks like this:

□

After I create the graphic, I save it to an unused letter space in the object file - in my case the small roman. As I mentioned above, I used the up arrow location to store the graphic

Now to run MERGE. Just answer the questions. Merge will ask the input file (the one you want to merge (or fill) to and the output file (the name for the resulting letters). Then you'll be asked for the character that contains the MERGE (or fill) graphic. I used the up arrow. YOU'LL ALSO BE ASKED FOR THE HIGHEST LETTER TO MERGE WITH - this is different than the other routines. You answer the LAST letter you want merged, usually the "z". MERGE and FILL ALWAYS START AT THE "!" CHARACTER.

Now wait a few seconds. The resulting file will be saved to disk. Unless your merge takes up the entire letterset frame, you should use option 13 - proportionalize. Then print your resulting letterset. Mine looks like this:



I can't believe I just got through that. Looking it over - it even makes sense! That is the MERGE routine. With it you can create or remove standard character patterns. Now that that is over I only have two more toughies to deal with. I'm not going to jump in though! I want you to spend a little time with this MERGE routine. Think it out. Get the feel. Give yourself some time to understand it and me some time for a drink and some thought before I have to crash into the MAGNIFY / EXPAND routine!

CHOICE 9, EXPAND / MAGNIFY

If you've been using LSMU from earlier versions of DOTWRITER, please read this carefully: it's entirely new and different.

This utility lets you increase the size of each character, horizontally, vertically, or both at once. You'll be shown the current Width, number of Lines, and Height of the letterset you choose, and then you'll be asked for the HORIZONTAL increase, in Dots. If the old width was 13, and you want to double the width, you should answer "13" (the total will be "26"). However, you can answer with a number that will give an odd size, a three-fold increase, etc. The only limit is what will fit into the new letterset frame and what fits on the screen. Of course, if you aren't increasing the size by an integer amount (2X, 3X, etc.), not all the dots will be magnified, and the resulting character will be somewhat distorted. We've found that this often produces nice-looking results, but not always.

Next, you'll be asked for the VERTICAL increase, also in dots: the same rules apply here as for horizontal. Note that, if you increase just the vertical size, the new characters will be tall and skinny; if you increase just the horizontal, they will be wide and fat; and if you double both of them, the resulting characters will be FOUR times the original size (we are dealing with areas here).

That's all there is to utility #9. If you've been using the earlier LSMU and miss the "screened" effect of "expansion", you can still get that effect by using TGEAP afterwards to just run the cursor through the characters to draw "blank" rows and columns.

That covers the MAGNIFY/EXPAND routine. Next we have to cover the strangest utility of all - WIGGLE. It lets you offset dot patterns in various combinations of sine waves. To be honest, I really don't understand all of its power! To make it easy for me to explain and to give you something to experiment with, I'll give you a brief explanation of the routine and a few examples. You will have to take it from there and EXPERIMENT. This time it is a cop out. I could load this manual with examples and still not have fully explained the function, because there are lots of variations possible. One thing I will say, though: the Wiggle routine works best with complicated or large graphics, not with simple lettersets like the ones used in this manual. For that reason, I will include a GEAP logo which I will distort in various ways to give you an idea of what you can do. The rest is up to you.

CHOICE 10, THE WIGGLE ROUTINE

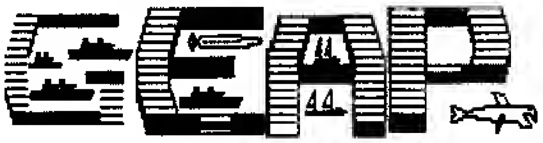
I haven't been looking forward to this discussion but I've always felt that the best way to do work you don't want to do was to jump in and get dirty. Actually, from a user standpoint, the WIGGLE routine is fairly simple. It's explaining it that is tough. I guess that the best way is to start with a normal graphic and then wiggle it a bit to let you see what happens. I'm not even going to try to give you total training. Most will have to be experimentation on your part. Let me see what I can do!

First, the explanation. The WIGGLE routine works best on graphics with full screen (128 x 6) dimensions. That is due to the nature of the routine. Essentially what we do is allow you to add a sine curve to the graphic. The wave effect can run from top to bottom or from left to right.

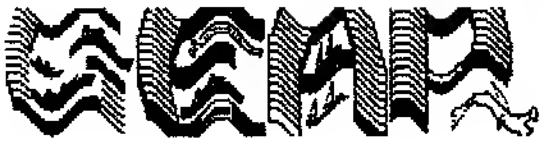
You will be prompted for the INPUT (source) and OUTPUT (result) file names as always. You' be asked for the range of characters to wiggle. You'll be asked for the kind or direction of wiggle you want ie. Up Down Left or Right.

That is the wiggle feature. I know it is a bit underdone, but I did warn you about it. I have tried to explain it more fully, but I've found that most people just want to play with it any way! The results are mixed. Some things look good and some don't. You just have to try it to see what happens.

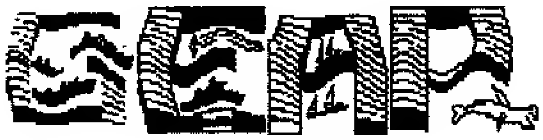
Here are some examples of the wiggle result - starting with:



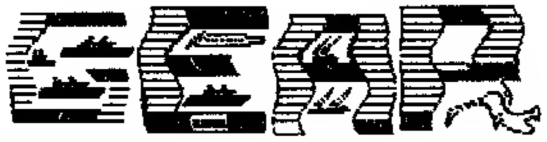
Now I'll wiggle it!



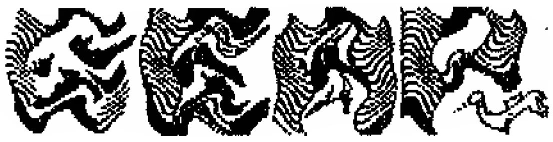
Now, a different wiggle.



A left / right taper results in the following distortion:



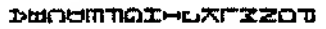
Here is another result!



I'm afraid that's about all I can tell you about the WIGGLE routine. Use it and try different values. It won't take long to catch on and the results can be worth it. Also, you can take heart in the fact that the worst of the utilities are over. The remaining ones are straight forward and you will have no more to puzzle over!

CHOICE 11, ROTATE / MIRROR IMAGE, SHRINK ROUTINE

Finally, another simple routine. You will be prompted, as always, for the INPUT and OUTPUT files, and the range of letters to be ROTATED. You will also be asked for the degrees that you want to rotate the object file. Since the TRS-80 (trademark of Tandy Inc.) graphics are not square, the 90 and 180 degree rotation produce the best results. Here is an example of the PL letterset rotated 90 degrees!



Now for the MIRROR letter option. You will be fully prompted for input, and the output will be a mirror image of the object letterset. As always, you should have a file prepared to accept the output from the routine. In the case of mirror image only, the output frame will be the same size as the input frame. In the case of rotation, read on.

There are a few warnings necessary. First, the routine rotates around the center of the frame so you should center the characters in the frame. Menu option 7 will do that for you. Also, The rotation takes room so be sure you have a big enough frame. The easiest way to do that is to create a large frame. Copy the object letterset into it and then center the resulting letterset. When the rotation is done. Use option 7 again and get the characters into the upper left corner of the frame. Finally, run option 12 and find the minimum storage size needed for the new letterset. Use option 3 to format that file and option 2 to copy the rotated letterset into the minimum frame. That's all folks!

Finally, the SHRINK routine. Shrinking a letterset, while it sounds fantastic, can be an aggravation. Often, they don't look like what you had in mind. Remember, we're working with a fixed pattern of dots and if we start subtracting dots we can eliminate the special effect created by proper dot placement.

If you want to shrink, just ENTER passed the other options and choose shrink. You'll be asked for the number of dots VERTICALLY and HORIZONTALLY that you want to shrink. Unless you're shrinking a solid block, be conservative. 1 or 2 dots is best.

Now, shrink - here is a sample result using the ROMAN letterset. First, standard ROMAN.

ABC DEF ghi jKl 123 !"T

Now, The result after the shrink rountine!

ABC DEF ghi jKl 123 !"T

That is it! Not too shabby - On to the next!

CHOICE 12, THE MINIMUM STORAGE ROUTINE

This little beauty will save you a lot of time and it is easy to use. Once selected, this routine asks you for the object file's name. It then asks for the range of characters. In most cases it is best to specify ALL of the letterset so the range would be "!" to "z". The routine then checks every character in the range, selects the largest one and then computes the smallest frame necessary to hold the largest character in the file. The result is printed on the screen.

With this routine, it is possible to work in a frame with PLENTY of room and still be able to reduce the frame size to minimum. It is best to have the character set left justified and moved all the way to the top of the frame if you want the smallest possible result.

Once you have the smallest frame parameters, use option 3 to create the file and option 2 to copy the letters into it. Another easy one isn't it?

CHOICE 13, PROPORTIONALIZE

This final option is NECESSARY for the new program users. The entering questions are the same as the other options. Tell it the name of the letterset you want to proportionalize. ALWAYS DO THE ENTIRE LETTERSET. Tell it the name of the output letterset (we use a filename with no extension when creating a letterset - like ROMAN - and add PR to the proportionalized version - like ROMAN/PR. Italicized versions are /IPR endings). Now just let it run! If you want to include the "hard space", be sure to include "127" in your range.

Option 13 looks at each letter, left justifies it, measures it and stores the width of the letter in a file at the end of the letterset. When DOTPRINT loads a letterset, it reads the WIDTH table and computes the proportion on the basis of the information received. This is how we can proportionalize the ALTERNATE font or fonts too large to be loaded into the computer!

* * * * *

That's "all" there is to the Letterset Manipulation Utilities -- but that's quite a lot. By now, it should be pretty obvious why people have been able to develop such good character sets and special effects with DOTWRITER: the combination of DOTPRINT, TGEAP, and LSMU gives you everything you need, from design and customization through super-fancy printing. We hope you will enjoy using it as much as we do!!

APPENDIX A

```
. S save figure to disk         .    F draw frame               .
. T transfer figure from disk   .    H display storage limits   .
. W write screen to printer     .    L list instructions        .
. A set dimensions for figure   .    E exit                     .
. . go to flashing .           .Arrow(s) move flashing -        .
.................................................................
............COMMANDS WHEN FLASHING .IS PRESENT...............
. Arrow(s) Move flashing . in indicated direction. Erases as   .
.            it moves.                                          .
. <SHIFT>+Arrow(s) Draw line in indicated direction.            .
. <CLEAR> Clear screen.                                         .
. - Return to flashing - sign.                                  .
. 0-7  Number keys control speed of flashing ..                 .
.................................................................
                   (Hit <ENTER> to continue)
```

INDEX